System Design As A Three-Phase Dual-loop (TPDL) Process: Types of knowledge-applied sources of feedback, and student development as independent learners

Professor Moshe Barak, Department of Science and Technology Education,
Ben Gurion University of the Negev, Israel

RESEARCH

Abstract

This study aimed at exploring how high school students deal with designing an information system, for example, for a small business or a medical clinic, the extent to which students develop as independent learners while working on their projects, and the factors that help or hinder fostering students' design skills. The three-phase dual-loop (TPDL) model for system design is proposed, according to which design consists of *conceptual design, structural design* and *detailed design*, and includes a *human-driven* feedback loop and an *instrumentation-driven* feedback loop. It was found that the design of a real-life system is a complicated task for high school students because it requires the integration of conceptual knowledge, primarily in the phase of defining a system's objectives and planning its general structure, and procedural knowledge, for example, in the phase of handling the detailed design, implementation and testing. The common situation in schools is that students learn and practice using procedural knowledge, whereas achieving conceptual knowledge is a long-term process. Therefore, it is essential to engage students in design tasks of increasing complexity from early stages in school in order to enable them to accumulate experience and construct their own knowledge about all phases of system design.

Key words

system design, information system, conceptual and procedural knowledge

Introduction

Teaching design concepts lies at the heart of technology and engineering in many countries including the United States, Britain, Australia and Israel. For example, in an international research study recently conducted by Hacker, De Vries and Rossouw (2009), technology education experts worldwide gave the highest average grades to the concepts of design (as a verb) and systems as being central to technology and engineering education. In high school, engaging students in design often takes place in the context of developing advanced technological systems in varied areas such as control systems, robotics and computer sciences. Moreover, learning design, by nature, is frequently project-based. Namely, the students learn design concepts by developing artifacts and systems aimed at solving particular problems or answering the needs of an individual or of society. Applying this teaching-learning method in school, however, is a rather challenging task for both the teachers and the students since project-based learning requires shifting most of the responsibility for learning from the instructor to the learner, which is unusual in conventional K-12 schooling. Moreover, as Fincher and Petre (1998) specifically mention, managing project work in computer sciences is particularly problematic because these projects are often complex, combining design, human communication, human-computer interaction and technology to satisfy human, organisational and technical demands. Nevertheless, many Israeli computer science teachers encourage their students to prepare a graduating project in their final year of high school. In these projects, students deal with the task of designing an information system for a customer, such as a small business or a medical clinic. This paper addresses a research study aimed at exploring how students deal with an open-ended design task, to what extent, if at all, engaging learners in a relatively complex assignment of system design contributes to their development as independent learners, and what are the factors that foster or impede this process.

Conceptual Framework

Learning to design and project-based learning in the light of learning theories

The psychological and educational literature of the past half a century has been greatly influenced by *constructivism* (Dewey, 1933; Piaget, 1950), and *social-constructivism* (Vygotsky, 1978; Palincsar, 1998) learning theories, claiming that individuals actively construct new knowledge, meaning and understanding from their experience and through social and cultural interaction, rather than obtaining knowledge from external resources, for example, the teacher. These learning theories have supported the development of reform-based instructional approaches in science and technology education, such as problem-based learning, project-based learning and design-based learning, which are closely associated with the current study.

Problem-based learning (PBL) is a learner-centered instructional approach in which students collaboratively conduct research, integrate theory and practice, and apply knowledge and skills to develop a viable solution to a problem (Savery, 1996; Savery and Duffy, 1995). The aim of this instructional method is not only to enhance content knowledge, but also to foster the development of general

competencies such as communication, problem-solving and self-directed learning skills.

Project-based learning and *design-based learning* are closely related to problem-based learning in that learning is organised around a central subject and is aimed at achieving a shared goal (Savery, 1996; Blumenfeld et al., 1991; Thomas, 2000). However, while problem-based learning has to do mainly with inquiry, in design- or project-based learning, learners are often expected to produce an end-product, for example, a model, an innovative artifact or a system aimed at solving a practical or real-life problem. The term design has two different meanings: as a verb or as a noun. As a verb, design refers to the process of originating, developing, implementing, evaluating and improving a product, a structure or a system. As a noun, design relates to the outcomes of the design process, for example, a plan, a sketch, a diagram or a model. Teaching design values both the learning process and its outcomes or products. Recently, researchers have demonstrated the advantages of using project-based learning and design-based learning in teaching subjects such as science (Krajcik et al., 1994; Mehalik et al., 2008), integrated programs for learning science and technology (Barak and Raz, 2000), electronics (Barak, 2005; Barak Shachar, 2008); and computer sciences (Scherz and Polak, 1999).

According to the constructivist view of learning, knowledge is constructed by the learner through cycles of experiencing and interacting with physical and social environments. Yet, what kinds of knowledge do people use and construct while dealing with system design? This point is discussed in the following sections.

Types of knowledge addressed in design projects

In the epistemological literature, it is common to distinguish between three main types of knowledge (Hiebert, 1986; McCormick, 1997, 2004):

- Declarative knowledge (also called propositional or factual knowledge) involves knowledge about facts, for example, knowing that a week includes seven days.
- Procedural knowledge refers to knowledge about how to do something or how to handle a specific task, for example how to calculate the current in an electric circuit, or how to write a computer program. This type of knowledge often consists of formal language, symbolic representations, rules, algorithms, procedures, techniques and methods.
- Conceptual knowledge involves knowledge about the interrelationships between basic elements within a larger structure, for example, categories, principles and models. Conceptual knowledge means understanding the relationships between items of knowledge, for example, understanding how concepts such as energy, system or feedback apply across biological and technological systems. By definition, conceptual knowledge cannot be learned by rote, but must be constructed by rich, thoughtful and reflective learning.

I will now discuss the role that procedural and conceptual knowledge play in system design.

The Three-Phase Dual-Loop (TPDL) system design model

In the literature, the process of designing a new artifact or system is often presented as consisting of the following stages (in various variations): 1) identifying a problem or a need; 2) researching and setting specifications for the required solution; 3) generating alternative solutions and choosing the optimal one; 4) planning; 5) implementing; 6) testing and evaluating; and 7) improving. However, educators have understood increasingly that design is not a linear process as it appears from this seven-stage list, but rather a more iterative process that often involves moving backwards and forwards from one point to another in the design process. If this is the case, the division of the design process into the specific detailed stages as mentioned above helps only a little in understanding how designers work or which types of knowledge they use in each phase. There is a need for a model that describes the principal (rather than itemised) design phases, the types of knowledge designers use, and the sources of feedback they receive during the design process. Such a model is illustrated in Figure 1.

The TPDL system design model shown in Figure 1 that was derived from the literature on system engineering and analysis and design philosophy (Blanchard and Fabrycky, 1998; Miettinen, 2008) presents system design as comprising three major phases and two feedback loops.

In the conceptual design phase, the designer is required to investigate the user's needs and desires, learn the properties of a system or artifact already in use, its structure, functions, advantages and limits, and identify the changes or improvements required. In this phase, the designer must also consider social, organisational or environmental issues, for example, aspects related to safety, risks, development time and costs. The title 'conceptual design' indicates that this phase involves using *conceptual knowledge*. As noted above, this has to do with broad concepts and interrelationships between basic elements of knowledge within a larger system, for example, energy, control and data analysis. The second phase, the structural design phase, involves designing the system's functional requirements and structure,

System Design As A Three-Phase Dual-loop (TPDL) Process: Types of knowledge-applied sources of feedback, and student development as independent learners

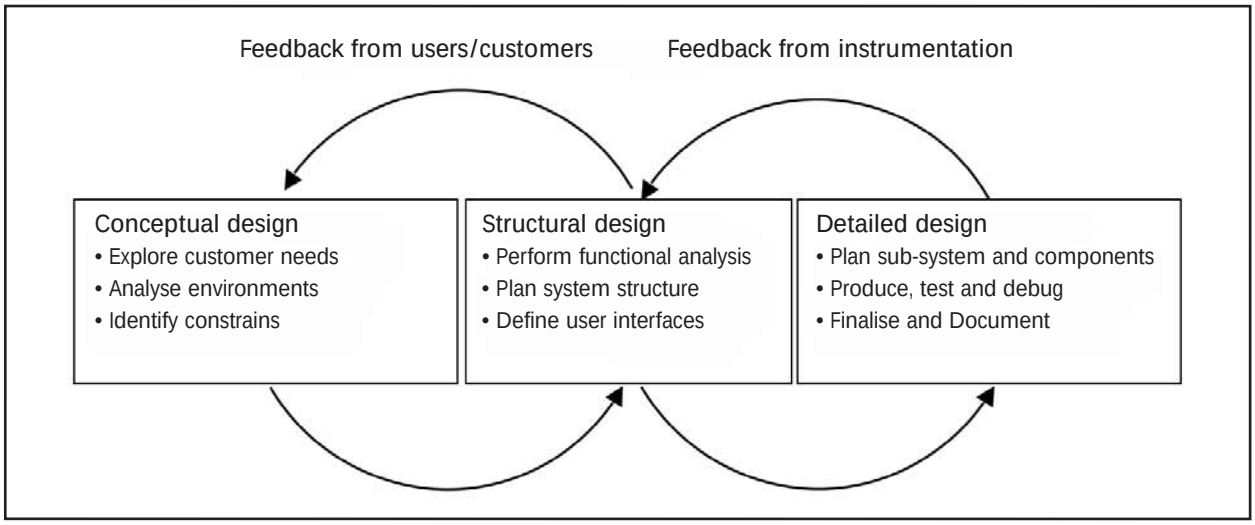


Figure 1. The Three-Phase Dual-Loop (TPDL) system design model

devising subsystems and choosing the major components, for example, energy source, type of controller or database structure in an information system. The detailed design phase consists of final planning, implementation, testing and improving the entire system and its components, for example, mechanical elements, electronic circuits or computer programs. This phase also involves finalising the system and preparing detailed documentation about the development process and outcomes. The detailed design phase primarily deals with *procedural knowledge*, for example, systematic planning of electronic circuits or mechanical systems and writing computer programs.

It is worth noting that the distinction between the conceptual, structural and detailed design phases and the types of knowledge required in each phase is not sharp and the borders are not exactly defined. Let us consider, for example, the case of designing and constructing a house. The architect is responsible for the house's conceptual design; the structural engineer deals with the structural design; the electrician, plumber and carpenter handle the detailed design. Yet, an architect must have basic knowledge in structural engineering, which is mainly of a procedural nature; the structural engineer should understand core architectural concepts; the electrician must know basic electricity theory, as well as procedures and standards in this field. Notwithstanding, this example highlights the difference between the three design phases included in the TPDL model and the different types of knowledge used in each phase.

The dual-loop nature of design

The TPDL model illustrated in Figure 1 includes two different design loops. One loop refers to the transition forward and backward between the system's conceptual design and its structural design. This loop could be marked as *human-driven* because it is about discussion, negotiation and providing feedback among people – customers and users. The second loop involves testing and improving a system from a scientific or technical viewpoint, for example, calculating, optimising or troubleshooting. Since, in the phase of implementation and testing, feedback comes mainly from the system's performance, namely devices, this loop could be referred to as an *instrumentation-driven* loop.

In summary, the TPDL model helps in identifying the three major phases of designing a new artifact or system, and highlights the use of conceptual versus procedural knowledge in the different design phases. This model also emphasises that design includes one loop of getting feedback from people, mainly customers and users, and another loop of receiving feedback from instrumentation.

In the following parts of this paper, I will show how the TPDL model described above could help in analysing the advantages and limits of teaching design in school, with a focus on the example of information systems design.

The case of information systems design

The literature describes an information system as being comprised of people, machines and methods organized to collect, process, transmit and disseminate data that represent a user (Yourdon, 2007). For example, an information system for a business application frequently handles different kinds of data about customers, suppliers, employees, stock management, accounting, etc. Developing an information system is a classical design

process, involving all the elements of the TPDL model, as will be demonstrated below.

As noted, the natural way of teaching design in school is by engaging students in design projects. Connelly and Begg (2006) point to the advantages of using principles derived from the constructivist epistemology to provide the learner with the knowledge and higher-order skills necessary to understand and perform database analysis and design effectively as professional practitioners. These authors suggest a range of guiding principles for designing such a learning environment:

- Allowing learners to choose an authentic project grounded in professional context.
- Encouraging learners to take responsibility for learning and be aware of knowledge construction.
- Allowing learners to develop their own process to reach a solution.
- Providing learners with the opportunity to appreciate and appreciate other perspectives.
- Providing opportunities for collaboration and interaction.
- Providing learners with feedback and support.
- Encouraging reflection in the class.

Research objectives and guiding questions

On the one hand, teaching system design in the project method is a natural platform for applying the constructivist learning environment. On the other hand, design is a complex task requiring the consolidation of conceptual knowledge, procedural knowledge, technical skills and social aptitude, and it is not easy for teachers to integrate the teaching of design into the project method in traditional schooling. Consequently, the main aims of this study were to explore:

1. How students deal with designing an information system.
2. To what extent students develop as independent learners while handling a challenging design task.
3. The factors that help or hinder fostering students' skills as independent learners and confident designers.

Methodology

Participants and setting

The participants in this study were 40 12th grade students (aged 17-18) from two Arab cities in northern Israel. Each student prepared an individual graduating project in information systems design and attended a final oral exam. More details on the content of the projects and the working stages are provided in the Findings section. All of the students who majored in computer sciences were considered to be high-achievers in their schools and studied concurrently advanced mathematics and physics courses. In these high schools, the students' mother tongue is Arabic, and in school, they learn Hebrew that they also use in daily life, as well as English. Out of 20 students in each class, there were three to four girls.

Both of the teachers were university graduates in computer sciences with over three years of experience in guiding students in preparing graduating projects in computer sciences. The principal investigator carried out this research with full cooperation and help from both teachers (one was studying towards an MSc degree at Ben Gurion University of the Negev).

Research approach and data collection methods

The research adopted a qualitative approach in order to expose as many learning patterns as possible, mainly the students' actions and thoughts during their work on the projects. McCormick (2007) discusses in detail the role of a qualitative classroom case study research in technology education. Researchers like Myers and Avison (2002) and Hazzan et al. (2006) specifically discussed the advantages of qualitative research in the areas of computer sciences and information systems development. These authors show that the qualitative approach helps in understanding social and educational phenomena from the point of view of the participants, and particularly its social and institutional context. The current study was a case study research that took place in a real-life environment and during regular school hours (McCormick, 2007). This increased the likelihood that the findings would be authentic and reflected the reality of implementing the project method in teaching information system design in Israeli high schools.

Data collection (by the teachers and the researcher) included:

1. Writing a diary about the students' work on their projects in the lab, for example, their questions, difficulties and progress in performing the task.
2. Keeping samples of students' work, for example, sketches, drawings and computer programs.
3. The teacher in one of the schools interviewed a sample of 12 students twice, six from each school: once in the middle of the school year and then upon completion of their projects. The teachers selected the interviewees to represent a sample of low, mid-level and high-achieving students in the project work. The students were interviewed individually or in pairs. During these 15-20 minute interviews, the students were asked questions such as how they worked on their projects, what difficulties they encountered, who helped them, and what they thought about preparing a project in computer sciences.

4. Observing the final oral exams the students attended on their projects (in one school), and holding discussions with the examiners and the students during and after the exam.
5. Administering an open-ended summative questionnaire to the students upon completion of their projects.
6. Collecting copies of 12 summative final reports that the students had prepared on their projects at the end of the school year and analysing their contents.

To avoid overloading this paper, the Findings section relates mainly to the first four items in the list above.

Findings

How the students selected their project topics

At the beginning of the school year, the teachers explained the aims of the project to the students and asked them to suggest their ideas for project topics. The teachers took notes about the way each student chose his/her project topic. Later, the researcher asked the students about this point in the interview. Of the 40 students, 33 decided to develop an information system on a topic that was of personal interest to them or their parents' vocation or business. Of these students, eight had already dealt with the same subjects in the small programming exercises they had prepared while learning programming in the 10th or 11th grade. Seven students had difficulties in coming up with their own ideas for a project and accepted the topic the teacher had suggested to them. In summary, the vast majority of students in both schools chose subjects that interested them personally, as demonstrated below:

- One student worked on an information system for a small garage where he was working part-time as an apprentice.
- Another student, who was taking driving lessons at the same time, decided to develop an information system for a driving school.
- A third student worked on an information system for her father's dental clinic.
- A fourth student developed an information system for a bank where his mother was working.
- Another student constructed an information system for a school library.
- A sixth student designed an information system for his family's clothing store.
- A seventh student decided to develop an information system for a travel agency because, as she said, "it would be interesting to deal with flights, hotels and international tourist attractions."

The fact that the students worked on topics that were meaningful to them very positively influenced their motivation to cope with the task. On the other hand, the teachers concluded that these topics were often too complicated for the students to learn in-depth. Once the students realised this, they settled for making only a partial design of their system, as will be clarified in the following sections.

How the students explored customers' needs

In the first stage of the design process, the students were required to investigate the customers' requirements and prepare a document called a 'Project Charter.' This document is based on a standard template consisting of the following elements:

1. Background of the project.
2. Customers' needs.
3. Changes or improvements required in the existing system.
4. Aims and objectives in terms of detailed measurable results.
5. Criteria of success.
6. Consequences of failure.
7. Constraints and factors that might limit the planning and how to accommodate them.
8. Possible risks, their probabilities, and how to overcome them.

This list demonstrates that system design starts with conceptual investigation and decision-making about the required product. Some of the students were very serious about investigating a customer's needs. For example, a student who developed an information system for a travel agency visited several travel agents and interviewed local employees. The student learned about the types of information handled by the travel agent, such as flights, hotels and car rentals. He also studied the interface screens the users require in working with the system, and the kinds of reports and printed documentation the system must provide. In contrast, other students only partially investigated the topic they were working on. Of the 40 students, ten prepared a Project Charter that sufficiently covered all the required points, 16 prepared a document that only partly answered the demands and needed improvement, and 14 students prepared a superficial or incorrect document. This situation has to do with the fact the students lacked experience in preparing a Project Charter or held some misconceptions about the aim of this phase of the system design, as discussed later in the paper. These findings indicate that the preparation of a formal document about the system's conceptual design and specifications was not an easy task for the students, and only some of them completed it satisfactorily. The educational implications of this are discussed below.

System Design As A Three-Phase Dual-loop (TPDL) Process: Types of knowledge-applied sources of feedback, and student development as independent learners

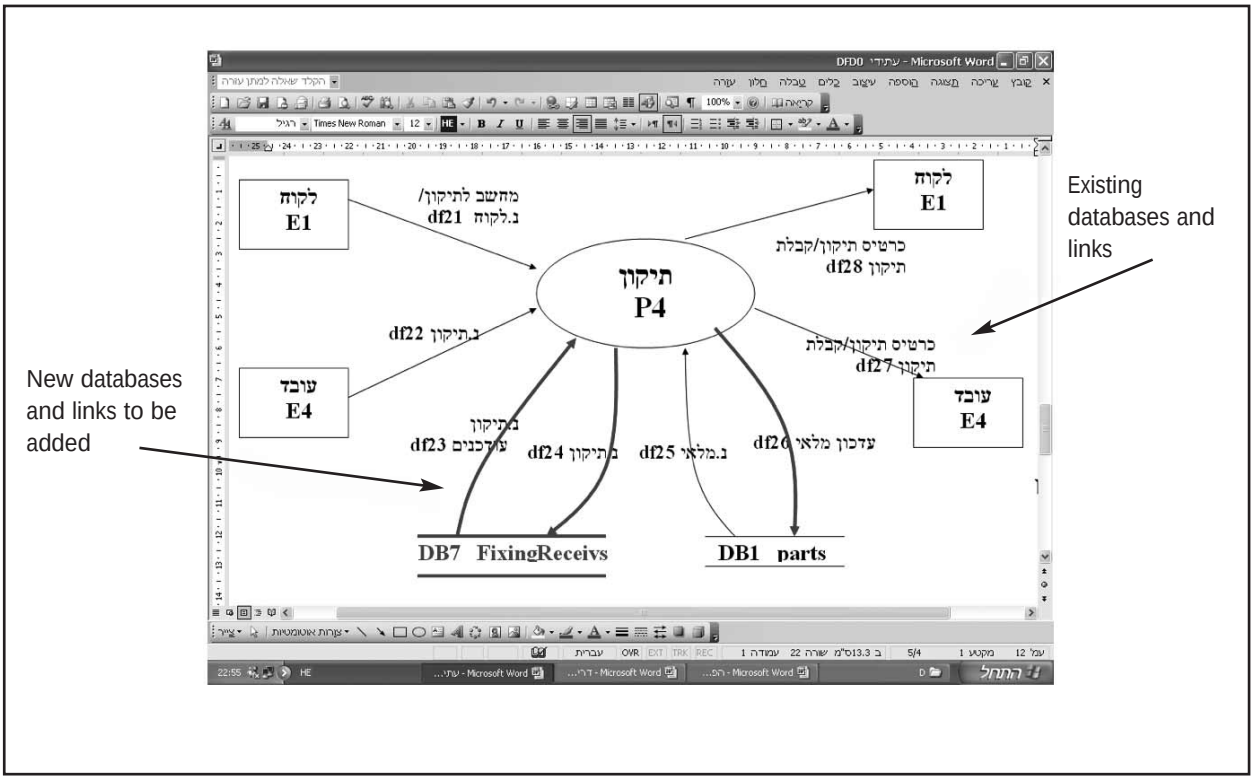


Figure 2. A Data Flow Diagram (DFD) sketched by a student as part of his information system design

How the students handled the information system's structural design

The planning of an information system takes place by drawing several diagrams and charts that present, for example, the architectural structure of the software system, its class inheritance (a way to form new classes that inherit the attributes and behaviour of pre-existing base classes), data flow, state machines (a behavioural model comprised of a finite number of states, transitions between these states, and actions), sequence diagrams, database tables and relationships. A systematic design of a software system is important not only for obtaining reliable and efficient software, but also for enabling communication between all individuals or teams involved in the system's development, implementation or future upgrade.

Figure 2 illustrates an example of a diagram called a Data Flow Diagram (DFD), which is a graphical representation of the flow of data through an information system. A student whose project was about enhancing an existing information system in a small computer store prepared this chart and included it in the booklet he prepared of his project.

The diagram shown in Figure 2 relates to process P4 – receiving a computer for repair in the store. This part of the system consists of two 'entities': E1 – the customer, and E4 – the store employee; the process is linked to two databases the student called DB1 – parts, and DB7 – FixingReceivs (student's spelling error). The thin arrows in Figure 2 illustrate databases and links that existed in the earlier system; the thick arrows indicate a new database (DB7) and links the student decided to add to it.

Drawing diagrams and charts for a system, as exemplified in Figure 2, was one of the main phases of a system's structural design. To do this, each student had to:

- Leave school to meet with the customers and visit their businesses.
- Learn the user's needs or habits in-depth.
- Understand the existing system in the organisation or business.
- Suggest priorities for updating or expanding the system.
- Draw Data Flow Diagrams for the existing system and the changes required, as illustrated in Figure 2.

The process described, in which the students actually planed their system by preparing several drafts of their design, lasted four-five weeks. The student sketched his chart in Figure 2 using the word processor drawing tool;

he then captured the entire screen using the Print Screen key and pasted the picture into his portfolio. Today, professional designers use various sophisticated design tools, called Computer Aided Software Engineering (CASE), which include design and debugging tools. In schools, however, it is common to do this work manually. As a result, at this stage of project work, the students proved to be fairly dependent on their teachers in checking their design and improving it.

Up to this point in the Findings section, we have seen that in the phases of the system's conceptual and structural design, the students were required to investigate the problem they were handling, learn the user's needs in-depth, prepare a formal document that comprises the objectives and anticipated specifications of the requested system, and plan the system's structure by drawing charts and graphs. Performing these tasks required the learner to deal mainly with conceptual knowledge, for example, understanding the user's needs, the types of information essential for the business, the structure and functions of an existing system, and how to enhance it. Good conceptual design also means that a designer identifies problems and comes up with ideas beyond what the user can point out.

How students carried out the detailed design

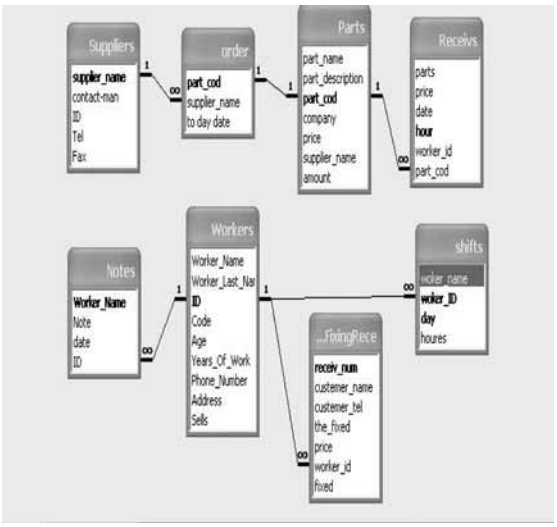
According to the TPDL design model, the detailed design phase includes planning, implementing, testing and improving a system's specific components and subsystems, as well as the entire system. In the current example, this phase involved mainly programming. This in itself is not a linear process but consists of cycles of writing, testing and improving computer programs.

The initial programming stages

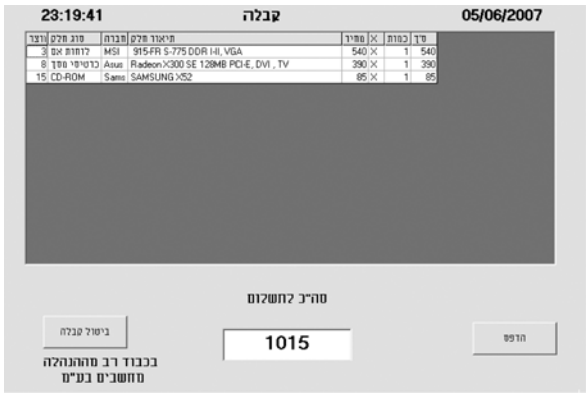
The students started the practical implementation by creating databases for the system using database software (Figure 3a), and programming all the logical functions, computations, user interface screens and reports using Visual Basic (Figure 3b).

As previously mentioned, when the students commenced their projects in 12th grade, it was after they had studied programming in C, Visual Basic and database software in 10th and 11th grades. However, the final project was the first time the students were required to integrate different programming tools and working environments in developing a system, rather than focusing each time on a specific method or programming language.

In the literature, it is common to include some basic guidelines for good programming (Kernighan and Plauger, 1974): using global variables sparingly; using library functions; replacing repetitive expressions by calling on a common function; identifying bad input data and recovering it if possible; ensuring that all the variables are initialised before use; using recursive procedures for



a. Typical database structure created by database software



b. Typical report by a customer created by Visual Basic

Figure 3. Examples of students' products in the implementation stage.

recursively defined data structures; and using data arrays to avoid repetitive control sequences.
Despite their previous experience, the students often ignored or violated these guidelines, as in the following example:

- One student developed a data system in which the user is required to enter his ID number (9 digits) in order to log in. This student did not include a verification clause in the program as to whether the user enters only digits between 0-9, and if the entire number is a valid ID number (there is procedure for this type of verification).
- After the teacher advised the student to include such a test in his system, he wrote a very cumbersome sequence and copied it into all of the forms that dealt with ID numbers, instead of building an efficient procedure for this purpose.

In the above example, the student did not follow two important programming rules: first, to check the data a user inputs before proceeding with any action; and second, the need to use modular programming for repetitive logical computational operations. Other frequent errors included the use of local versus global variables and variable types. In the interviews, students had the following comments:

"I am used to working this way."
"I didn't feel it was important."
"I didn't know it was required that I use procedures."
"The teacher didn't ask me to do that."

It is worth noting that although the Visual Basic working environment includes tools for debugging the kinds of errors shown above, the students often ignored these error messages in their initial programming steps and opted to resolve them by using the trial-and-error method. In fact, reading English messages is not easy for many Israeli students; for Israeli Arab-speaking students in particular, as in the current case, English is their third language after Arabic and Hebrew.

During this four-six week period, the students progressed very slowly, and many felt very frustrated. Below are some examples of comments made by students during this period:

"The computer went completely crazy."
"I don't know what the problem is."
"The program worked in theory but did not run properly on the computer."

Students' progress in programming
So far, we have seen that when the students started programming their system most encountered many problems. However, after the initial 'shock' many students experienced when starting to program their system, things improved gradually, as demonstrated below.

- Upon the teachers' strong recommendation, and with their help, the students moved to more modular programming methods rather than repeatedly using specific sections of a code in different parts of their program.
- The students improved their handling of Visual Basic error messages. For example, some students prepared a list of error messages they were receiving and translated them into Arabic or Hebrew.
- The students recognised that many of their mistakes were about defining and using variables, transferring parameters to procedures, etc. Consequently, many students included in their portfolio a table of all the variables they were using, including the name, type and function of each.
- Some students improved or re-wrote parts of programs they had prepared earlier to make them more efficient or elegant.

In discussions with students who had nearly completed their projects, they had the following reflections on their work:

"I now really understand the error."
"I made the same error again because I was in a hurry."
"I am writing down every error massage… I am taking notes… the next time it appears I will solve it faster…"

Indications of students' self-learning
An important question that guided this study was to what extent did the students use books or other resources or independently learn new subjects in computer sciences while working on their projects. To this end, the teacher asked the librarian in one of the schools to record each visit of the students in the school library, including the student's name, the visiting date and the names of books the students used or borrowed. Out of the 20 students, six visited the library frequently, while the others came only occasionally. Although the library has books on all of the subjects the students learned in school, the students were only interested in computer science books. They also often learned from the booklets of projects prepared by students in this school in previous years. The librarian commented that although the students were in their 12th grade in high school, some of them were visiting the library for the first time due to the project work.

In a summative discussion with three students in the second school, they had the following comments:

"In my project, I ran statistics and showed the results in diagrams. No one else had done something similar."

"In my project, the reports are very elaborate and not standard. I used a report generator that I had learned by myself."
"I used an updated book of new Microsoft tools; I also use ADO technology rather the older DAO one."

At this stage, the students were able to help each another on technical issues. For example, many students exchanged specific algorithms, helped their friends in solving particular programming requirements, or cooperated in designing a rich user interface.

The findings presented above highlight that the students progressed significantly, not only regarding the cognitive side of the programming, for example, planning, problem-solving and debugging, but also in their ability to think meta-cognitively during their work and reflect on their learning. In addition, the students internalised the advantages of helping each other in working on their projects rather than learning individually and even competing with one other, as is often found in conventional schooling.

The examiner's viewpoint and final scores

The researcher attended the final oral exams held for a class of 20 students for about four hours, talked informally with the examiner and the students, and documented the entire event. The examiner asked each student to demonstrate the system he/she had developed, explain specific parts of the programs, and describe the system's entire development process. Sometimes he asked the students to make some changes or improvements in the system and gave them 15-30 minutes to do so. In a discussion with the examiner, he had the following comments:
"I am looking to see if a student understands the structure of the information system he/she has constructed and the process of storing and updating data in the system."
"High school students are not professional programmers… it is enough that they acquire basic programming skills. It is more important to see if the students understand the process of information systems development, are able to explain what they are doing, and be aware of the advantages and limits of the system they have constructed."
"One cannot compare what the students have learned from working on their projects to what they have learned in a conventional course."
"Every year I am amazed at the level of knowledge many students exhibit. Their motivation is very high."

The examiner's comments demonstrate that in the eyes of professionals, fostering students' competencies relating to system design is more important than teaching specific programming skills in the class. Therefore, in Israeli schools the students' final grades on their projects (on a scale of 0-100), which are included in the matriculation certificate each student receives from the Ministry of Education, comprise an average of grades granted to each student from the teacher and the external examiner. In the current case, while the external examiners in the two schools based their grades primarily on the oral exam and the booklet each student prepared on his/her project, the teachers also took into account the students' efforts and progress throughout the project work. Of the 40 students, 40% were granted a grade ranging from 90-100 (excellent), 15% from 80-89 (very good), 25% from 70-79 (good), and 20% from 60-69 (poor). These findings, which are typical of Israeli schools, indicate that only part of the students excel in the graduating project on information systems. While most of the students progressed significantly in all areas regarding the system's implementation, mainly programming and debugging, the mission of the system conceptual design remained an unresolved issue for many learners.

Discussion

This study addressed the case of teaching design in the context of information systems development. The participants were high school students (12th grade) who prepared individual projects as part of majoring in computer sciences. As mentioned in the Conceptual Framework section, teaching design- and problem-based learning are aimed at creating a constructivist learning environment in school, in which the student constructs knowledge and develops cognitive and social skills by handling authentic problems and developing a new artifact or system aimed at answering the needs of an individual or of society. The main aim of this study was to explore how students deal with the relatively challenging task of designing an information system. More specifically, the study sought to investigate the extent to which students develop as independent learners while handling challenging design tasks and identify the factors that help or hinder the fostering of students' learning skills.

Earlier in this paper (Figure 1), we observed the TPDL model for system design process, according to which design takes place in three main phases: conceptual design, structural design and detailed design. While design frequently starts by addressing conceptual issues, the final design phases are concerned essentially with applying procedural knowledge. We have also seen that while in the conceptual design phase, designers get feedback

essentially from individuals, namely customers, in the detailed design stage, which includes implementing and testing, feedback comes mostly from the system itself, namely instrumentation. Let us see how this model helps us in understanding the current research findings.

As presented in the Findings section, in the classes observed in the current study, a marked gap existed between the way the students dealt with the first stages of project work – mainly conceptual design and structural design, and how they managed the more advanced phases of the project's detailed design, including programming, testing and debugging.

At the beginning of the project work

When the students started their projects, they faced many difficulties and depended heavily on the teachers' help. Several factors caused this situation. One problem was that the students had relatively little experience with conceptual design, essentially investigating into the real-life needs or the customers' requirements, and writing realistic specifications for the system required. As already noted, this phase of the system design involves merely conceptual knowledge that individuals can gain through considerable experience and rich, thoughtful and reflective learning.

A second factor was in the transition from the system's conceptual design to its structural design due to the fact that the learners had only little experience in drawing charts related to information systems design. Moreover, since the students had previously learned computer sciences for two years exclusively in a computerised interactive environment (database software, Visual Basic), they considered the 'manual work' of the system planning, for instance drawing Data Flow Diagrams, as being marginal. For these students, the essence of the project was the 'doing,' namely the programming, rather than the planning.

A third factor, and likely the most critical one, which impeded the classes at the beginning of the project work was the situation whereby only the teachers could check the students' work, provide feedback, or help each learner individually. As indicated in the TPDL model, in the transition from conceptual design to structural design, designers receive feedback primarily from other individuals. In real life, designers get feedback from customers or potential users of the desired system. In the school situation, students have only restricted time and limited experience in learning in-depth a system aimed at a small business or a clinic. As a result, the students depend heavily on the teachers' help and approval of their conceptual and structural design. This situation is summarised in the Conceptual Design row shown in Table 1.

Towards accomplishing the project

After the students prepared the structural definition and plans for their systems they were developing, they moved to the detailed design phase. In the Findings section, we have seen that although many students faced difficulties at the beginning of the system's implementation, they progressed well in independently carrying out the programming testing stage, and improving their systems. This change could be related to several factors:

- After preparing the structural design, the students became familiar in general with the structure of the system they were developing, for example, the number and types of databases, the data flow, and the logical functions in the system. Therefore, the project became more concrete to the students.
- The programming and debugging work was relatively easier for the students because these actions involved

Design Phase		Real Life	School
Conceptual Design (investigating needs)	Who designs?	Expert	Student
	What is the main feedback source?	Customer, user	Teacher
Detailed Design (planning, implementing, testing)	Who designs?	Professional (considerable experience)	Student (little experience)
	What is the main feedback source?	Instrumentation	Instrumentation

Table 1. Conceptual design and detailed design in real life and in school

applying procedural knowledge primarily (at least at the level of the projects the students were preparing).
- The students had learned and practiced writing and testing similar computer programs in 10th and 11th grade prior to working on the more comprehensive project.
- Programming and debugging are iterative processes in which the designer gets immediate feedback from the computer rather than depending on the teacher's help.
- As noted in the Findings section, the students learned to use more effectively the system's error messages and debugging tools.
- At this phase of the project work, the students could collaborate and help one another more.

In summary, the students became much more independent learners in accomplishing their projects. This did not just involve gaining more experience but was the result of a change in the nature of the task. In addition, the students received feedback from the computer rather than depending on the teacher, and they applied their considerable previous experience in doing similar work.

Concluding remarks

The design of a real-life system is often a complicated task for high school students because it requires the integration of conceptual knowledge, mainly in the phase of defining a system's objectives and general structure, and procedural knowledge, for example, in the detailed design, implementation and testing phase. The common situation in schools is that students learn and practice mainly procedural knowledge, whereas accumulating conceptual knowledge is a long-term process. Indeed, the current research highlighted that the conceptual design phase might appear to be one of the most challenging parts for students inexperienced in carrying out a design project, because this task has to do mainly with exploring peoples' needs, desires and expectations. As students proceed to the structural design and detailed design phases, including implementing, troubleshooting and improving, the task becomes more concrete to them since it is essentially about procedures and instrumentation. Our recommendation is to engage students in small design tasks of increasing complexity from early stages in school, for example in junior high school or in the first year of secondary school, in order to enable them to accumulate experience and construct their own knowledge about all phases of system design. This is an essential preparation process before engaging students in broader real-life design tasks. After all, learning to design is closely related to the constructivist view of learning, which emphasises that individuals construct their knowledge and develop intellectual skills through cycles of experimentation, interaction with instrumentation, collaboration with peers and reflection.

The author would to thank Mr. Naif Awad for his considerable contribution to this study.

References

Barak, M. (2005) 'From order to disorder: The role of computer-based electronics projects on fostering of higher-order cognitive skills.' *Computers and Education*, 45(2), 231-243.

Barak, M. & Raz, E. (2000) 'Hot-air balloons: Project-centered study as a bridge between science and technology education.' *Science Education*, 84(1), 27-42.

Barak, M. & Shachar, A. (2008) 'Project in technology and fostering learning skills: The potential and its realisation.' *Journal of Science Education and Technology*, 17(3), 285-296.

Blanchard, B.S. & Fabrycky, W.J. (1998) *Systems engineering and analysis*, Englewood Cliffs, NJ: Prentice-Hall.

Blumenfeld, P.C., Soloway, E., Marx, R.W., Krajcik, J.S., Guzdial, M. & Palinscar, A. (1991) 'Motivating project-based learning: Sustaining the doing, supporting the learning.' *Educational Psychologist*, 26 (3 & 4), 369-398.

Connelly, T.M. & Begg, C.E (2006) 'A constructivist-based approach to teaching database analysis and design.' *Journal of Information Systems Education*, 17(1), 43-53.

Dewey, J. (1933) *How We Think*, (2nd ed.), New York: D.C. Heath,

Fincher, S. & Petre, M. (1998) 'Project-based learning practices in computer science education.' *Proceedings of the Frontiers in Education Conference* (FIE'98), November. 4-7, 1185-1191.

Hacker, M., De Vries, M. & Rossouw, A. (2009). *International research study on Concepts and Contexts in Engineering and Technology Education* (CCETE), New York, Hofstra University.

Hazzan, O., Dubinsky, Y., Eidelman, L., Sakhnini, V. & Teif, M. (2006) 'Qualitative research in computer science education.' *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*, March 1-5, 408-412.

System Design As A Three-Phase Dual-loop (TPDL) Process: Types of knowledge-applied sources of feedback, and student development as independent learners

Hiebert, J. (Ed.). (1986). *Conceptual and procedural knowledge: The case of mathematics*. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Kernighan, B.W. & Plauger, P.J. (1974) 'Programming style: Examples and counterexamples.' *ACM Computers Survey*, 6(4), 303-319.

Krajcik, J.S., Blumenfeld, P.C., Marx, R. W. & Soloway, E. (1994) 'A collaborative model for helping middle grade science teachers learn project-based instruction.' *The Elementary School Journal*, 94(5), 483-497

Myers, M.D. & Avison, D.E. (2002) *Qualitative Research in Information Systems: A Reader*. London: Sage Publications.

McCormick, R. (1997) 'Conceptual and procedural knowledge.' *International Journal of Technology and Design Education*, 7(1-2), 141-159.

McCormick, R. (2004) 'Issues of learning and knowledge in technology education.' *International Journal of Technology and Design Education*, 14(1), 21-44.

McCormick, R. (2007) 'Classroom case studies.' In: H. Middleton (ed.) *Researching Technology Education: Methods and Techniques*. International Technology Education Studies (3). Rotterdam, Netherlands: Sense Publishers, 6-27.

Mehalik, M.M., Doppelt, Y. & Schunn, C.D. (2008) 'Middle-school science through design-based learning versus scripted inquiry: Better overall science concept learning and equity gap reduction.' *Journal of Engineering Education*, 97(1), 1-15.

Miettinen, K. (2008) 'Design: Structure, Process, and Function: A Systems Methodology Perspective.' In P.E., Vermaas, P.A., Kroes, A., Light & S. Moore (eds.), *Philosophy and Design, From Engineering to Architecture*, Springer Netherlands, 217-231.

Palincsar, A.S. (1998) 'Social constructivist perspectives on teaching and learning.' *Annual Review of Psychology*, 49(1), 345-375.

Piaget, J. (1950) *The Psychology of Intelligence*, New York: Routledge.

Savery, J.R. (1996) 'Overview of problem-based learning: definitions and distinctions.' *The Interdisciplinary Journal of Problem-based Learning*, 1(1), 9-20.

Savery, J.R. & Duffy, T.M. (1995) 'Problem-based learning: An instructional model and its constructivist framework.' In B. Wilson (ed.), Constructivist learning environments: *Case studies in instructional design*, Englewood Cliffs, NJ: Educational Technology Publications, 135-148.

Scherz, Z. & Polak, S. (1999) 'An organizer for project-based learning and instruction in computer science.' *ACM SIGCSE Bulletin*, 31(3), 88-90.

Thomas, J.W. (2000) *A review of research on project-based learning*, Autodesk, San Rafael, CA. Retrieved from http://www.bie.org/files/researchreviewPBL.pdf.

Vygotsky, L.S. (1978) *Mind in Society*, Cambridge, MA: Harvard University Press.

Yourdon, E. (2007) *Structured Analysis*, Retrieved January 2009, from http://www.yourdon.com/jesa/jesa.php.

mbarak@bgu.ac.il

RESEARCH